Journal of Al-Nahrain University Vol. 9(2), December, 2006, pp.73-77 Science

Cryptanalytic Attack on Merkle-Hellman Knapsack Based on Diploid Genetic Algorithm

[1]Dr. Sarab M. Hameed
Computer Science Department, College of Science, University of Baghdad

Abstract

Cryptanalysis is the science and study of methods of breaking ciphertext. Spillman presented a Genetic Algorithm (GA) for cryptanalysis of Merkle-Hellman knapsack. Then Garg et al improved the Spillman GA. The contribution of this paper is to harness the power of GA used by Garg et al to get more reliable results and in less computation time. The threefold objectives (reliability, efficiency, and implementation simplicity) are satisfied in this paper by utilizing a GA, known as diploid GA, which has two advance operators *diploid representation scheme* and *dominance map*. Diploid GA is used to translate each number in ciphertext into the correct knapsack assignment which represents the ASCII code for the plaintext characters. Results are compared with Garg et al and they proved that diploid GA is more efficient and highly successful in finding the correct bit pattern for the hard knapsack sum

Introduction

Cryptography is the science and the study of secret writing and it is practitioner by cryptographer. Cryptanalysts are practitioner of cryptanalysis, the science and study of method of breaking ciphertext [1, 2].

There are various types of cryptanalytic attacks based on the information known to cryptanalyst [2, 3]. In *ciphertext-only attack*, the cryptanalyst has the ciphertext of several messages all of which have been encrypted using the same encryption algorithm. The cryptanalyst's job is to recover the plaintext of as many messages as possible.

In *known plaintext attack*, the cryptanalyst has access not only to the ciphertext of several messages, but also to the plaintext of those messages. His job is to deduce the key(s) used to encrypt the message or an algorithm to decrypt any new message encrypted with the same key.

In *chosen-plaintext attack*, the cryptanalyst not only accesses to the ciphertext and associated plaintext of several messages, but he also chooses the plaintext that gets encrypted. His job is to deduce the key(s) used to encrypt the messages or an algorithm to decrypt any new message encrypted with the same key(s).

In *adaptive- chosen plaintext attack*, is special case of a chosen plaintext attack. Not only can the cryptanalyst choose the plaintext that is encrypted, but he also modifies his choice based on the results of pervious encryption.

The first application of Genetic Algorithm (GA) in the cryptanalysis of knapsack cipher is suggested by Spillman in 1993 [4]. The goal of this GA is to translate each number into correct knapsack, which represents the ASCII code for the plaintext characters.

Then, in 2006 Garg P. et al improved the efficiency of Spillman's genetic algorithm attack on knapsack cipher [5]. They take certain restrictions on the encoding algorithm. These are:

1. Only the ASCII code will be encrypted.
2. The super increasing sequence will have eight elements.
3. Plaintext has not more than 100 characters length.

They concluded [5] that the efficiency of the GA attack can be improved, even more than Spillman GA, by variation of the initial assumptions like crossover operation, mutation, and size of population. The one-point crossover operation applied in the algorithm. The mutation process moves between two random points. The great size of population, high crossover probability, and small mutation probability are the most optimal arrangements for GA. The results are worse when the size of the population and probability of crossover operator decreases, and when the coefficient mutation increases. Wrong coefficients destroy the chromosomes that are well-selected.

This paper presents an approach to cryptanalyst the Merkle-Hellman algorithm and all the information available to cryptanalyst is ciphertext only. The approach uses mechanism of diploid GA to recover the plaintext from ciphertext without knowing the key. Our contribution in this work is threefold. By utilizing diploid GA with two advance operators *diploid representation scheme* and *dominance map* in addition to the traditional GA operators, we satisfy the following.

- Reliability: Get more accurate results.
- Efficiency: The accurate results are produced with much faster time (i.e. with less computation time).
- Implementation Simplicity: The GA code is still easy to implement and require few parameters to tune.

We begin this paper by illustrating Merkle-Hellman algorithm. Then, we discuss how to use diploid GA to cryptanalyst the algorithm and recover the plaintext. Finally, we present our results and conclude this work.

1. Merkle-Hellman Knapsacks

The first algorithm for generalization public key encryption was the knapsack algorithm developed by Ralph Merkle and Martin Hellman. The Merkle-Hellman algorithm is based on knapsack problem. The knapsack problem posed a set of positive integers and a target sum, with the goal of finding a subset of the integer that summed to the target. The knapsack problem is NP-complete, implying that to solve it probably requires time exponential in the size of the problem (i.e. the number of integers) [1, 6].

Merkle-Hellman knapsack encodes a binary message as a solution to knapsack problem, reducing the ciphertext to the target sum obtained by adding terms corresponding to the 1s in the plaintext. That is, blocks of plaintext will be converted to knapsack sum by adding into sum those terms that match with 1 bit in the plaintext as illustrated in the following example:

Plaintext	1 0 1 0 0 1	0 1 1 0 1 0
Knapsack	1 2 5 9 20 43	1 2 5 9 20 43
Ciphertext	1 5 43	2 5 20
Target Sum	49	27

A knapsack is represented as a vector of integer terms in which the order of the terms is very important. There are actually two knapsacks: an easy one, to which a fast algorithm exists, and the hard one, derived by modifying the elements of the easy knapsack.

Merkle-Hellman algorithm uses this property. The private key is a sequence of weights for superincreasing (i.e. set each of whose elements is larger than the sum of all pervious elements) knapsack problem. Public key is a sequence of weights for hard knapsack problem and it is created from private key by multiply all of the values by number *n, mod m*. The modulus (*m*) should be greater than the sum of all the numbers in the sequence, and *gcd (n,m)=1*, gcd represents the greatest common divisor.

To encrypt a binary message, first break it up into blocks equal to the number of items in the knapsack sequence. Then, allowing a *one* to indicate the item is present and *zero* to indicate that the item is absent, compute the total weights of the knapsacks one for every message block.

To decrypt the ciphertext, a legitimate recipient of the message knows the private key, as well as n and m. The recipient must first determine n^{-1}, multiply each of the ciphertext value by $n^{-1} \bmod m$, and partition with the private key to get plaintext value. n^{-1} is determined by equation (1) [1].

$$n^{-1} = n^{\phi(m)-1} \bmod m \qquad (1)$$

Where $\phi(m)$ is the number of elements in the reduced set of residues modulo m (i.e. the number of positive integers less than m that are relatively prime to m).

2. The Cryptanalytic Approach

The presented cryptanalyst utilizes diploid GA to cryptanalysis knapsack cipher. Cryptanalyst has only cipher text and want to recover plaintext from it. The ciphertext forms as integer number that represents target sum of hard knapsack problem.

The usefulness of diploid is that it allows for memory of previously discovered solutions which may be useful as the environment change. In a diploid GA, a pair of identical coding chromosomes, *called homologous* pair, represents an individual. Each binary gene was described by two genes, a *modifier* gene and *functional* gene. The functional gene took on normal 0 or 1 values. The modifier gene took on value M or m. Each of these pairs must be decoded into single string said to represent an actual problem solution. The values in this string are said to be the *expressed* values associated with the diploid chromosome [7].

The mechanism of eliminating the conflict between homologous chromosomes through a genetic operator is called *dominance*. At a locus, it has been observed that one allele (the dominant allele) take precedence over (dominates) the other alternative alleles (recessive) at that locus.

Diploid individual Representation and Initialization

Diploid GA work on a population of individuals. The population represents a set of candidate solutions to the problem at hand. The individual is triallelic and its values are 0, 1, and -1. The number of bits in each individual is equal to the number of elements key. Here, the individual length is equal to eight. Figure 1 depicts individual representation.

1	-1	0	1	1	1	-1	0
1	0	0	0	1	1	-1	-1

Figure 1: Individual Representation

At the beginning of the diploid cycle, a random number of individual are created that represent population size pop_{size}

Fitness Function

To evaluate the individual, first we apply Hollstien rule to control dominance that say both 1 and -1 map to 1, but 1 dominates 0 and 0 dominates -1. Figure 2 depicts this dominance scheme. With this scheme the more effective allele becomes dominant, thereby shielding the recessive [7].

	0	-1	1
0	0	0	1
-1	0	1	1
1	1	1	1

Figure 2: Triallelic Dominance Map

Then, evaluate the expressed value using fitness function given by equation (2) [4]. The expressed value of figure 1 will be as figure 3.

1	0	0	1	1	1	1	0

Figure 3: Expressed Value

$$fitness=\begin{cases} 1-\left(\dfrac{Target-Sum}{Target}\right)^{1/6} & \text{if } Sum<Target \\ 1-\left(\dfrac{|Target-Sum|}{MaxDiff}\right)^{1/6} & \text{if } Sum>Target \end{cases} \qquad (2)$$

Let be an arbitrary solution and the public key $A=\{a_1,a_2,...,a_n\}$, then:

$$Sum=\sum_{j=1}^{n} a_j m_j$$

(3)

$$Target=\sum_j a_j$$

(4)

$$FullSum=\sum_{j=1}^{n} a_j$$

(5)

Maxdiff=max {Target, FullSum-Target}

Diploid GA Operators

After defining the representation of individual solutions and computing the fitness function for each individual, the diploid GA operators are applied to get better solutions. These operators are selection, crossover, and mutation.

Given a population of individuals, each one with a fitness value, the algorithm progresses by randomly selecting two for mating. The binary tournament selection is used [7, 8]. In tournament, a pair of individuals is competed. The individual with a high fitness value is copied into the mating pool. That is, individuals with a high fitness value have a greater chance of being selected to generate children for the next generation. This process is repeated until the mating pool is filled with number of individuals equal to population size (pop_{size}).

Then, two stages process of crossover operator is applied. In the first stage of diploid information combination, crossover occurs between the homologous chromosomes within a single individual to form single stranded gametes. This stage is called *gametogenesis*. After gametogenesis, a haploid gamete from one parent is combined with a haploid gamete from another parent to form the homologous pair for a diploid offspring's chromosome. This stage is called *fertilization*. Figure 4 demonstrates crossover operator. The crossover operator operates with probability P_c.

After the new generation has been determined, the individual are subjected to a low rate mutation function. The mutation converts each allele type to one of the other types with equal probability. Thus, mutation accounts to changes in value (i.e. from a 1-valued allele to a 0 and vise versa) and changes in dominance (i.e. from dominant 1 to recessive -1) [9, 10]. The mutation operates with probability P_m.

Diploid GA Solution

When applying diploid GA operators, a new generation is created. The process of applying diploid GA is repeated until stopping criteria is met. Here, the stopping condition is when the fitness function equal to 1 which means that we reach to exact solution the represent the ASCII code of the character in the plaintext (i.e. recover plaintext).

Results

The diploid GA was tested using the same example of [6]. The Merkle-Hellman algorithm was used with the following parameter:

1. *Private key* (easy knapsack): (1 3 7 13 26 65 119 267 504 1007 2013 4027 8053 16107 32213).
2. m= 65423.
3. n = 21031 and we compute n^{-1} = 5363.
4. *Public key* (hard knapsack): (21031 63093 16371 11711 23422 58555 16615 54322).

After determining the parameters of the algorithm, the encryption process is applied on plaintext" MACRO" and the following ciphertext is obtained 65728 37646 108739 103130 128821 each number represents a sum of the difficult knapsack.

The ciphertext is attacked by cryptanalyst using diploid GA with parameters: pop_{size} =5, P_c = 0.9, and P_m = 0.1 to translate each number in ciphertext into the correct knapsack assignment which represents the ASCII code for the plaintext characters.

he cryptanalyst obtains the result in ASCII code. That is, it correctly identified the enciphered characters which represent the plaintext. The result is: 10110010 10000010 11000010 01001010 11110010 which represent the ASCII code for MARCO. Table 1 and 2 illustrates the results of Garg et al, and our results.

Table 1: Garg et al Results

Character	Average population size
M	17
A	112
C	271
R	89
O	87
Average	115

Table 2: Our Results

Character	Population
M	5
A	5
C	5
R	5
O	5
Average	5

Since diploid GA embodies a form of temporal memory that is distributed across the population, then it is highly successful in finding the correct bit pattern for the hard knapsack sum.

Comparing our approach that uses diploid GA to cryptanalysis knapsack cipher with that of Garg et al that uses simple GA, we can draw the following encouraging remarks:

- Our diploid GA approach is more efficient than Garg et al. The reasons behind this result are the following. First, the diploid GA works with pop_{size} equal to 5, while Garg et al GA has to work with, on average, 115 individuals. In other words, about 95.65% of computation time per every generation is saved when diploid GA is adopted. Second, when the adopted diploid GA converges to the accurate solution for the first number in ciphertext (i.e. ASCII code of the first character in the plaintext) which is normally occurred when the fitness function equal to 1, it continues to re-evolve this final population according to the next number in the ciphertext and so on with out requiring to re-initialize the population randomly from the beginning as this is the case of the Garg et al GA approach. This re-evolving ability comes from the fact that diploid GA has a diverge number of solutions saved besides those converged solutions that expressed when population continues to evolve toward a given solution. However, the GA of Garg et al evolves and thus converges its population towards only one solution, and to re-change solution requires re-initializing population instead of re-evolving it.
- Code implementation of the diploid GA is still as easy as the traditional GA.

Conclusion

This paper presents an approach that utilizes diploid GA to cryptanalysis Merkle-Hellman knapsack. The approach is simple and efficient and offers a powerful tool to cryptanalysis knapsack cipher that successfully finds the correct bit pattern for the hard knapsack sum (i.e. ASCII code for plaintext characters). Our approach is more efficient than Garg et al because their methods use average 115 population to give the correct result while in our approach, the pop_{size} is set to 5.

References

[1] D. Denning "Cryptography and Data Security", Addison Wesley, 1982.

[2] B. Schneier. "Applied Cryptography, Algorithms, Protocols and Source Code in C", second edition, John Wiley & Sons, 1996.

[3] W. Stallings " Cryptography and Network Security: Principles and Practice", second edition, Prentice-Hall. Inc., 1999.

[4] Spillman R., "Cryptanalysis of Knapsack Ciphers Using Genetic Algorithms", Cryptologics, 17(4):367-377, 1993.

[5] Garg P., Shastri A., and D.C. Agarwal, "An enhanced Cryptanalytic Attack on Knapsack Cipher Using Genetic Algorithm", Transaction

on Engineering, Computing and Technology, vol. 12, 2006.

[6] C. P. Pheeger, S. L. Pfleger, "Security in Computing", third edition. Prentice-Hall, Inc, 1997

[7] D. E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison Wesley, 1989.

[8] M. Mitchell "An Introduction to Genetic Algorithm" MIT press, 1996.

[9] R. E. Smith "An Investigation of Diploid Genetic Algorithms for Adaptive Search of Non stationary Functions", M. Sc. Thesis, Alabama, Tuscaloosa, 1988.

[10] R. E. Smith and D.E. Goldberg, "Diploidy and dominance in Artificial Genetic Search", Complex Systems, vol. 6, 1992, pp 251-285

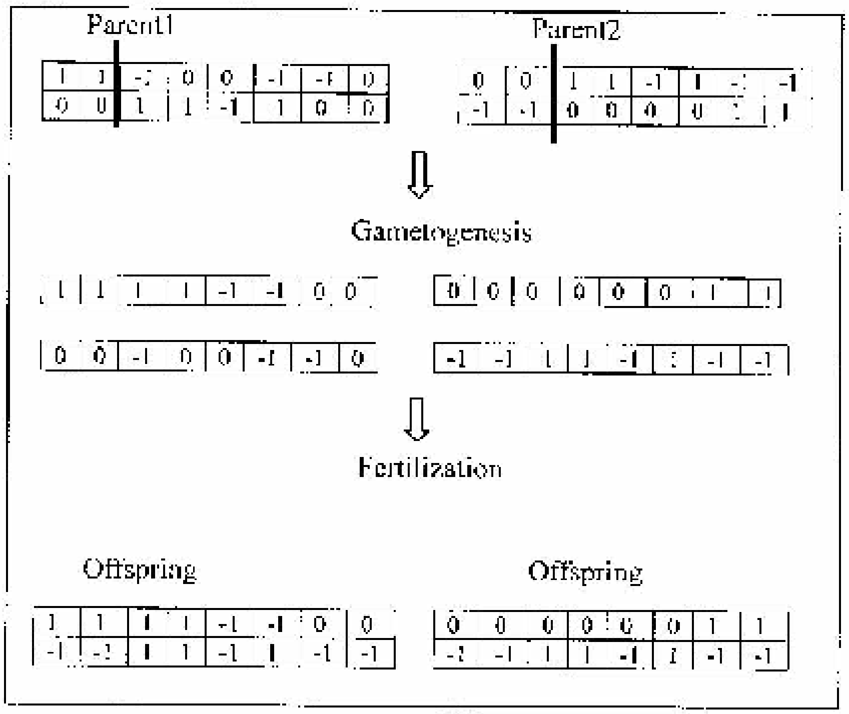

Figure 4: Crossover: Gametogenesis and Fertilization

المستخلص

تحليل الشفرة هو علم ودراسة طرق كسر الشفرة. سبلمن قدم طريقة لتحليل حقيبة ميركل هيلمن باستخدام الخوارزمية الجينية. كلارك وآخرين حسن الخوارزمية الجينية لسبلمن. هذا البحث يساهم في تقوية الخوارزمية الجينية المستخدمة من قبل كلارك للحصول على نتائج ذات موثوقية عالية وبوقت قليل . تم تحقيق ثلاثة اهداف وهم الموثوقية والكفاءة وبساطة التطبيق باستخدام الخوارزمية الجينية الثنائية التي لها عاملان متقدمان صيغة التمثيل الثلاثي وسيطرة الهيمنة. استخدمت الخوارزمية الجينية الثنائية لترجمة كل رقم في النص المشفر الى الحقيبة الصحيحة التي تمثل ASCII Code الى احرف النص الصريح. تم مقارنة النتائج مع كلارج وآخرين. اثبتت النتائج بأن الخوارزمية الجينية ناجحة جداً في ايجاد نمط القطعة الصحيح لمجموع الحقيبة الصعب.